Программа курса

Создание web-приложений с использованием Python

Продолжительность курса: 184 пары

Расписание: 2 пары аудиторных занятий два раза в неделю и минимум 6 часов на выполнение домашних заданий

Порядок чтения предметов курса может быть реализован двумя путями:

- последовательно – HTML/CSS, JavaScript, Python;
- сначала HTML/CSS; затем, после завершения курса HTML/CSS, начинаем курс Python – два раза в неделю; начиная с Модуля 4 курса Python, один раз в неделю – JS и один раз – Python; после завершения JS, продолжаем курс Python – два раза в неделю.

Требования к поступающим:

- возраст от 15 до 55 лет;
- уровень подготовки: уверенное владение ПК.

Тематический план

1. **Разработка веб-страниц на языке разметки HTML5 с использованием каскадных таблиц стилей CSS3 . . . . . 14 пар**
2. **Разработка клиентских сценариев с использованием JavaScript . . . . . 30 пар**
3. **Создание web-приложений с использованием Python . . . . . 140 пар**

Разработка веб-страниц на языке разметки HTML5 с использованием каскадных таблиц стилей CSS3

Версия 3.0.1

Продолжительность курса: 14 пар

Цель курса

Обучить слушателя созданию и верстке статических web-страниц с использованием технологий HTML5, CSS3. Сложить для слушателя целостное представление о технологической цепочке создания web-сайтов и сформировать понимание актуальных тенденций развития web-технологий. Научить слушателя выбирать наиболее подходящий способ для создания web-страниц. Научить тестировать и проверять код web-страниц.

По окончании курса слушатель будет:

- знать и уметь применять основы HTML – теги, атрибуты и способы структурирования содержимого web-страниц для создания форматированных документов;
- знать и уметь применять основы CSS – значения, списки, цвета, шрифты и другие метрики форматирования;
- владеть навыками проверки и отладки кода web-документов;
- владеть навыками формирования содержимого web-документов для различных экранов – от стандартных браузеров до мобильных устройств;
- владеть навыками быстрого и качественного форматирования сложных web-документов;
- знать основы HTML5 и CSS3.

В качества редактора для создания можно использовать любой бесплатный продукт. Например, Notepad++, Microsoft Visual Studio Community.

По окончании данного курса студент сдает практическое задание и теоретический экзамен по материалам курса. Для допуска к экзамену должны быть сданы все домашние и практические задания.

Студент должен создать web-сайт с последующим размещением в Internet.
Основные требования: блочная верстка, валидный код.

Тематический план

Модуль 1. Введение в Web-технологии. Структура HTML. Форматирование текста при помощи HTML . . . . . **2 пары**

Модуль 2. Форматирование при помощи CSS. Списки. CSS отступы и поля . . . . . **3 пары**

Модуль 3. Графика в web-дизайне. Оптимизация графики. Гиперссылки. Принципы навигации web-сайта . . . . . **3 пары**

Модуль 4. Таблицы. . . . . **2 пары**

Модуль 5. Позиционирование. Верстка web-страниц блоками. . . . . **2 пары**

Модуль 6. Формы. Фреймы . . . . . **2 пары**

Модуль 1

Введение в Web-технологии. Структура HTML. Форматирование текста при помощи HTML

1. **Введение в предмет.**
2. **Введение в языки разметки. Язык разметки гипертекста HTML.**
   - Internet.
   - Протокол HTTP.
   - Развитие HTML, версии. Версия HTML5.
   - Вопросы межбраузерной совместимости. Война браузеров.
   - W3C.
3. **Теги – основной элемент структуры HTML. Правила записи тегов и их атрибутов в стандарте HTML5. Синтаксические отличия HTML4, XHTML, HTML5.**
4. **Основные ошибки в записях тегов.**
   - Спецификации <!DOCTYPE HTML>.
   - Валидация документа при помощи FireFox – дополнение HTML Validator.
   - Понятие well-formed.
   - Прародители HTML5: SGML и XML.
5. **Структура HTML5 документа.**
   - Основные элементы и их назначения.
   - Новые теги задания структуры: <header>, <nav>, <section>, <article>, <aside>, <footer>. Доступность новых тегов в современных браузерах. Отображение новых тегов в устаревших браузерах.
6. **Кодировки страницы и теги <meta>.**
   - Применение тега <meta> – задание информации о странице (expires, refresh, autor, copyright, keywords, description).
   - Задание кодировки страницы при помощи тега <meta>.
   - Символьные подстановки и кодировки.
7. **Классификация тегов: линейные и блочные.**
   - Линейные.
   - Блочные.
8. **Модель форматирования текста: заголовки и абзацы. Элементы <p>, <h1>...<h6>. Выравнивание текста в блочных элементах: атрибут align.**
9. **Классификация тегов: логическое и физическое форматирования.**
   - Теги физического форматирования.

- Теги логического форматирования.
- Краткий обзор основных тегов логического форматирования: <abbr>, <acronym>, <cite>, <code>, <del>, <dfn>, <ins>.

10. Практика: создание простейшей web-страницы.

Модуль 2

Форматирование при помощи CSS. Списки. CSS отступы и поля

1. **CSS – каскадные таблицы стилей.**
   - Введение. Обзор версий. Назначение: HTML служит для задания структуры, CSS – для форматирования.
   - Встраивание CSS в HTML при помощи атрибута style.
     Правила записи CSS свойств.
2. **Теги без форматирования – блочный, – линейный.**
3. **Аналогия HTML и CSS на примере линейных и блочных тегов.**
4. **Дополнительные свойства CSS для форматирования текста: letter-spacing, line-height,text-indent, text-transform, white-space, word-spacing.**
5. **Использование атрибутов class и id для задания стилей.**
   - Создание стилей для тегов, классов, идентификаторов внутри тега <style>. Понятие селекторов. Правило записи селекторов: селектор тегов, селектор классов, селектор идентификаторов, универсальный селектор *.
   - Приоритет использования стилей (tag / class / id / style). Повышение приоритета правилом !important.
   - Наследуемость стилей. Стандартные значения свойств.
   - Отслеживание стилей при помощи средства разработки firebug (дополнение для Firefox).
6. **Использование внешних CSS файлов стилей.**
   - Подключение CSS файлов при помощи тега <link> и инструкции @import.
   - CSS файлы и кэш браузера.
7. **Практика: форматирование текста при помощи CSS.**
8. **Создание списков.**
   - Неупорядоченные списки: элементы <ul>, <li>.
   - Упорядоченные списки: элементы <ol>, <li>.
   - Атрибуты type, value, start.
9. **Создание вложенных списков.**

10. **Форматирование списков при помощи CSS.**
    - Свойства list-style-type, list-style-image, list-style-position.
    - Сокращенная запись свойства list-style.
    - Оформление многоуровневых списков. Вложенные селекторы.
11. **Списки определений: элементы <dl>, <dd>, <dt>.**
12. **Управление отступами и полями.**
    - Свойство margin и его потомки margin-left, margin-top, margin-right, margin-bottom.
    - Свойство padding и его потомки padding-left, padding-top, padding-right, padding-bottom.
    - Отличие padding от margin и их назначения.
    - Отмена отступов по умолчанию у некоторых тегов: <body>, <h1>...<h6>, <p>.
13. **Практика: создание списков.**

Модуль 3

Графика в web-дизайне. Оптимизация графики. Гиперссылки. Принципы навигации web-сайта

1. **Форматы графических файлов в Web.**
2. **Тег <img /> и его атрибуты (src, alt, width, height, border).**
    - Свойство border – аналог атрибута border.
    - Задание свойств margin, padding, border для изображения.
    - Выравнивание изображений на странице при помощи атрибута align. Аналог атрибута align – свойство float.
3. **Фон страницы – свойство background.**
    - Задание фона в виде цвета: background-color. Обязательное задание фона для элемента <body>.
    - Задание фона в виде изображения: background-image, background-repeat, background-position, background-attachment.
    - Изображения и кэш браузера.
4. **Общие сведения о гиперссылках.**
    - Тег <a> и его атрибуты (href, target).
    - Эргономика, удобство навигации.
5. **Абсолютная и относительная адресация.**
    - Организация внешних ссылок.
    - Организация внутренних ссылок с помощью элемента <a>. Атрибуты id и name.

- Организация «смешанного» перехода (на указанный элемент во внешнем HTML-документе).
- Графические ссылки. Отмена границ у ссылок.

6. **Создание меню при помощи структуры списков (<ul>, <li>), его форматирование. Свойство display. Преобразование ссылки в блочный элемент.**
7. **Псевдоклассы.**
   - Псевдоклассы ссылок: active, hover, link, visited.
   - Псевдоклассы для обычных элементов: first-child, first-line, first-letter.
8. **CSS свойство cursor.**
9. **Практика: работа по разработке галереи изображений.**
10. **Свойства из CSS3.**
    - Работа с фоном: создание градиентов, изменение размеров фона – свойства background и background-size.
    - Работа с границами: скругленные края у блоков – свойства border-radius.
    - Задание полупрозрачности элементам страниц – свойство opacity.
    - Полная поддержка селекторов CSS 2.1.
11. **Работа с мультимедиа.**
    - Вставка видео на странице посредством тега <video>.
    - Вставка аудио на странице посредством тега <audio>.
    - Создание изображений и анимации посредством тега <canvas>.
    - Использование SVG формата.

Модуль 4

Таблицы

1. **Создание простейшей таблицы. Теги <table>, <tr> и <td>.**
   - Атрибуты border, cellspacing cellpadding. Их возможные аналоги CSS: border, padding.
   - Указание ширины и высоты ячейки: атрибуты width, height. Правила задания ширины и высоты. Аналоги CSS: свойства width, height.
   - Выравнивание данных в таблице: атрибуты align и valign.
     Аналоги CSS: свойства text-align, vertical-align.
   - Управление цветом фона и цветом рамок таблицы (отдельной строки, отдельной ячейки).
   - Использование изображений в качестве фона таблицы (отдельной строки, отдельной ячейки).

2. **Объединение ячеек: атрибуты colspan, rowspan.**
3. **Теги логического структурирования таблиц: <thead>, <tbody>, <tfoot>. Теги логического группирования столбцов: <colgroup>, <col>.**
4. **Управление рамками таблицы: атрибуты frame, rules.**
5. **Практика: создание сложных таблиц.**
6. **Основы табличной верстки. Пример табличной верстки: ее минусы.**

Модуль 5

Позиционирование. Верстка web-страниц блоками

1. **Свойство position.**
   - Рассмотрение позиционирования: relative и absolute.
   - Свойства top, left, bottom, right.
2. **Свойства visibility, overflow.**
3. **Практика.**
4. **Основы верстки блоками. Правила верстки.**
   - Вложение блоков.
   - Задание ширины и высоты блокам при помощи свойства width и height.
   - Обтекание блоков. Отмена обтекания блоков. Свойства float и clear.
   - Правила задания отступов и полей.
   - Задание минимальной высоты и ширины блока: свойства min-height, min-width. Задание этих свойств в браузере IE6.
   - Выравнивание внутри блоков (margin, text-align, line-height, position). Кроссбраузерность выравниваний.
5. **Рассмотрение простейших структур страниц и элементов.**
   - Структура сайта фиксированного размера.
6. **Резиновая структура. Блоки с отрицательными margin.**

Модуль 6

Формы. Фреймы

1. **Введение в формы.**

2. **Управляющие элементы форм.**
   - Кнопки (отправки, сброса, пр.).
   - Флажки.
   - Кнопки с зависимой фиксацией (радиокнопки).
   - Всплывающие списки.
   - Текстовый ввод.
   - Выбор файлов.
   - Скрытые управляющие элементы.
3. **Создание форм при помощи HTML.**
   - Элемент <form>.
   - Элемент <input>.
   - Элемент <button>.
   - Элементы <select>, <optgroup> и <option>.
   - Элемент <textarea>.
   - Метки <label>.
   - Структура форм: <fieldset> и <legend>.
4. **Элементы форм из HTML5.**
5. **Валидация форм при помощи HTML5.**
6. **Форматирование элементов форм при помощи CSS.**
7. **Фреймы и их структура (теоретические сведения).**
   - Тег <iframe>.
   - Использование фреймов для подключения внешних ресурсов (YouTube, Google Maps и т. д.).

Разработка клиентских сценариев с использованием JavaScript

Версия 2.0.0

Продолжительность курса: 30 пар

Цель курса

Обучить слушателя разработке клиентских сценариев с использованием JavaScript. Научить выбирать правильные механизмы и конструкции для решения той или иной задачи.

По окончании курса слушатель будет:

- владеть базовыми конструкциями языка JavaScript, такими как переменные, условия, циклы, строки, массивы функции и т. д.
- знаком с ООП и его основными понятиями;
- уметь обрабатывать возникающие ошибки;
- разбираться в понятиях «событие», «обработчик события».
- создавать функции-обработчики различных событий;
- понимать отличия BOM и DOM;
- уметь взаимодействовать с объектами из BOM и DOM;
- разбираться в тонкостях реализации клиентских сценариев под разные браузеры;
- владеть принципами создания форм и анализа данных пользователя с использованием регулярных выражений;
- уметь сохранять пользовательские данные с помощью механизма cookie;
- понимать особенности применения HTML5 по отношению к JavaScript;
- уметь сериализовать и парсить данные, используя JSON;
- владеть принципами создания асинхронных запросов при помощи AJAX.

По окончании данного курса студент сдает все практические задания курса. На основании всех сданных заданий выставляется оценка по предмету.

Тематический план

Модуль 1. Введение в JavaScript ..... **2 пары**

Модуль 2. Объект. Массивы. Объект Array. Строки. Объект String. Объект Date. Объект Math. Введение в ООП ..... **6 пар**

Модуль 3. Обработка событий ..... **4 пары**

Модуль 4. Browser Object Model. Document Object Model ..... **4 пары**

Модуль 5. Формы ..... **2 пары**

Модуль 6. Проверка достоверности форм. Использование Cookie ..... **2 пары**

Модуль 7. JSON, AJAX ..... **4 пары**

Модуль 8. Введение в jQuery ..... **6 пар**

Модуль 1

Введение в JavaScript

1. **Сценарии, выполняемые на стороне клиента.**
2. **Что такое JavaScript?**
3. **История создания JavaScript.**
4. **Различия между JavaScript и Java, JScript, ECMAScript.**
5. **Версии JavaScript.**
6. **Понятие Document Object Model.**
7. **Понятие Browser Object Model.**
8. **Внедрение в HTML документы. Редакторы кода JavaScript.**
9. **Тег <noscript>.**
10. **Основы синтаксиса.**
    - Регистрозависимость.
    - Комментарии.
    - Ключевые и зарезервированные слова.
11. **Переменные. Правила именования переменных.**
12. **Типы данных.**
13. **Операторы.**
    - Арифметические операторы.
    - Операторы отношений.
    - Логические операторы.
    - Оператор присваивания.
    - Битовые операторы.
    - Приоритет операторов.
    - Оператор typeof.
14. **Ввод/вывод данных. Диалоговые окна.**
15. **Условия.**
    - Что такое условие?
    - if
    - if else
    - Тернарный оператор ?:
    - switch

16. Циклы.

- Что такое цикл?
- while
- do while
- for
- break
- continue
- Понятие метки.

17. Что такое функция?

- Синтаксис объявления функции.
- Параметры функции.
- Возвращаемое значение функции. Ключевое слово return.

18. Объект arguments.

- Цель и задачи объекта.
- Свойство length.

19. Область видимости переменной. Ключевое this.

20. Рекурсия.

Модуль 2

Объект. Массивы. Объект Array. Строки. Объект String. Объект Date. Объект Math. Введение в ООП

1. Объекты.

- Что такое объект?
- Введение в объектный тип данных.
- Объект Object.
- Ключевое слово new.
- Понятие свойства.
- Добавление свойств. Синтаксис добавления свойств.
- Синтаксис обращения к свойствам.

2. Массивы.

- Что такое массив?
- Объект Array.
- Создание массива.

- Обращение к элементам массива.
- Свойства и методы Array.

3. **Строки.**
   - Объект String.
   - Свойства и методы String.
4. **Задержки и интервалы. Периодический вызов функций.**
5. **Объект Date. Обработка даты и времени.**
6. **Объект Math. Свойства и методы. Случайные числа.**
7. **Что такое ООП?**
8. **Три фундаментальных принципа ООП.**
   - Инкапсуляция.
   - Наследование.
   - Полиморфизм.
9. **Понятие класса и объекта в терминах JavaScript.**
10. **Свойства.**
11. **Методы.**
12. **Свойства-акссесоры.**
    - get-свойства (геттеры).
    - set-свойства (сеттеры).
13. **Конструктор.**
14. **Понятие prototype.**
    - Что такое prototype.
    - Цели и задачи prototype.
15. **Наследование.**

Модуль 3

Обработка событий

1. **Что такое событие?**
2. **Что такое обработчик события?**
3. **Обработка событий в сценариях.**
4. **Управление стилями элементов web-страницы.**
5. **Объект event и его свойства.**

6. **Обработчики событий по умолчанию (стандартные обработчики), запрет вызова стандартного обработчика.**
7. **Объект Image. Управление рисунками и ролловерами.**

Модуль 4

Browser Object Model. Document Object Model

1. **Что такое Browser Object Model?**
2. **Объекты Browser Object Model.**
   - Объект Window. Открытие, перемещение и изменение размера окон.
   - Объект Navigator. Управление браузером.
   - Объект Screen. Свойства экрана.
   - Объекты Location и History. Перемещение по страницам.
   - Коллекция Frames. Управление фреймами.
3. **Что такое Document Object Model?**
4. **Отличия DOM от BOM.**
5. **Представление HTML-документа в виде дерева.**
6. **Объекты модели DOM. Иерархия узлов.**
7. **Свойства и методы модели DOM. Модель событий DOM.**
8. **Изменение дерева DOM.**
9. **Знакомство с объектами Document и Link.**
10. **Управление выделением и текстовым диапазоном: объекты Selection и TextRange.**
11. **Особенности DOM в HTML5.**

Модуль 5

Формы

1. **Применение форм. Размещение элементов формы в HTML.**
2. **Коллекция Forms. Создание и программирование элементов формы.**
   - Кнопки: элементы Button, Submit, Reset.
   - Текстовые поля: элементы Text, Password, File Upload, Textarea.
   - Скрытое поле формы: общее понятие об элементе Hidden.

- Флажок: элемент Checkbox.
- Переключатель: элемент Radio.
- Список: элементы Select, Option.

Модуль 6

Проверка достоверности форм. Использование Cookie

1. **Объект RegExp. Правила записи регулярных выражений.**
2. **Методы объектов String и RegExp для работы с регулярными выражениями.**
3. **Проверка достоверности данных формы.**
4. **Что такое cookie?**
5. **Преимущества и недостатки cookie.**
6. **Создание, использование и удаление cookie.**

Модуль 7

JSON, AJAX

1. **Что такое JSON?**
2. **Цели и задачи JSON.**
3. **Синтаксис JSON.**
   - Переменные.
   - Объекты.
   - Массивы.
4. **Объект JSON.**
   - Что такое сериализация?
   - Что такое парсинг?
   - Методы stringify и parse.
5. **Настройка пользовательской сериализации в JSON. Метод toJSON.**
6. **Синхронные и асинхронные запросы.**
7. **Что такое AJAX?**

8. **Объект XMLHttpRequest.**
    - Создание через ActiveX объект.
    - Создание через объект XMLHttpRequest.
9. **Методы и свойства XMLHttpRequest.**
10. **Понятие HTTP заголовка.**
11. **Использование метода GET. URL кодирование.**
12. **Использование метода POST.**

Модуль 8

Введение в jQuery

1. **Что такое jQuery?**
2. **Цели и задачи jQuery.**
3. **История создания jQuery.**
4. **Версии jQuery.**
5. **Подключение jQuery.**
6. **Доступ к элементам страницы при помощи функции $.**
7. **Понятие селектора.**
8. **Типы селекторов.**
    - CSS селекторы.
    - jQuery селекторы.
9. **Взаимодействие с DOM.**
    - Создание новых элементов DOM.
    - Вставка элементов DOM.
    - Передвижение элементов DOM.
    - Копирование элементов DOM.
    - Взаимодействие с атрибутами.
    - Traversing. Методы обхода DOM. Метод filter, next, nextAll, prev, prevAll, siblings и др.
10. **События и jQuery.**
    - Создание обработчиков событий с использованием jQuery.
    - Удаление обработчиков событий.
    - Практические примеры.

11. **Анимация и jQuery.**
12. **AJAX и jQuery.**
    - JSON.
    - Механизмы AJAX внутри библиотеки jQuery.
    - Использование метода GET.
    - Использование метода POST.
    - События и AJAX в рамках jQuery.
    - Обработка ошибок.
13. **Использование jQuery плагинов.**
    - Понятие плагина.
    - Подключение плагина.
    - Примеры плагинов.

Создание web-приложений с использованием Python

Версия 1.0.4

Продолжительность курса: 140 пар

Цель курса

Обучить слушателя разработке веб-приложений с использованием Python.

По окончании курса слушатель будет:

- разбираться в тонкостях построения веб-приложений с использованием Python;
- понимать особенности реализации механизмов ООП в Python;
- обрабатывать и анализировать данные форм;
- использовать стандартные возможности Python;
- применять регулярные выражения;
- понимать принципы функционального программирования;
- сохранять данные пользователя в файлах cookies;
- работать с механизмом сессий;
- понимать принципы сетевого взаимодействия;
- взаимодействовать с источниками данных;
- внедрять AJAX в веб-приложения;
- уметь пользоваться системой контроля версий;
- понимать основы командного взаимодействия;
- применять паттерны проектирования;
- использовать юнит-тестирование;
- понимать и разбираться в тонкостях паттерна MVC;
- создавать веб-проекты с использованием Python и паттерна MVC;
- использовать Flask/Bottle;
- разрабатывать web-приложения с помощью фреймворка Django.

По окончании данного курса студент сдает практическое задание и теоретический экзамен по материалам курса. Для допуска к экзамену должны быть сданы все домашние и практические задания. Практическое задание должно охватывать максимум материала из различных разделов курса.

Перед началом данного предмета необходимо предоставить студентам доступ к следующему курсу Microsoft Imagine Academy: Introduction to Python.
Этот курс является подготовкой к экзамену MTA 98-381: Introduction to Programming Using Python.

Тематический план

Модуль 1

Введение в web-программирование на Python

1. **Введение в web-программирование и принципы работы web-приложений.**
   - Основы технологии клиент-сервер.
   - Обзор протокола HTTP, знакомство со структурой Request и Response.
   - Обоснование и истоки возникновения web-программирования. Отличия серверного web-программирования от клиентского. Цели, задачи, направление развития, краткая история.
   - Принципы и этапы загрузки web-страницы.
2. **Обзор языков программирования.**
   - Знакомство с основными парадигмами программирования.
   - Обзор современных языков программирования.
   - Понятие алгоритма.
   - Знакомство с языком Python, сферы применения.
   - Введение в Python. Интерпретатор Python и его окружение.
3. **Введение в Python.**
   - Понятие интерпретатора и порядок установки.
   - Знакомство со средами программирования:
     - стандартный пакет программирования IDLE и Python Shell;
     - IDE PyCharm, Spyder, Visual Studio;
     - Atom.
4. **Типы данных, переменные и синтаксические конструкции.**
   - Тип и значения.
   - Переменные как объект в языке Python.
   - Имена переменных и зарезервированные слова.
   - Инструкции.
   - Операторы и операнды.
   - Приоритеты операторов.
   - Операции над переменными.
   - Порядок выполнения программы.
   - Ввод/вывод.
   - Преобразование типов.
   - Ошибки синтаксические и логические, работа с ними.

Модуль 2

Операторы ветвлений, циклы, исключения

1. **Условные инструкции и их синтаксис.**
   - Понятие «блока» выполнения.
   - Логические выражения и операторы.
   - Операторы ветвления if ... else.
   - Вложенные конструкции.
2. **Понятие исключений.**
   - Типы исключений.
   - Перехват исключений.
   - Особенности работы с try ... except.
3. **Циклы.**
   - Понятие итерации.
   - Цикл while.
   - Бесконечные циклы.
   - Управляющие операторы continue, break и else.
   - Цикл for.
   - Локальные и глобальные переменные.

Модуль 3

Строки, списки

1. **Строки.**
   - Кодировка ASCII, Unicode, UTF-8, Byte-code.
   - Строка – неизменяемая последовательность символов.
   - Методы строк.
   - Особенности работы со строками.
   - Срез строки.
   - Экранированные последовательности.
   - «Сырые строки».
   - Форматированный вывод.
   - Модуль string.
   - Байты и кодировки.
   - Регулярные выражения, модуль re.

2. **Списки.**
    - Понятие классического массива.
    - Понятие коллекции объектов.
    - Ссылочный тип данных list.
    - Создание списков.
    - Генераторы списков.
    - Работа со списками.
    - Методы списков.
    - Оператор принадлежности in.
    - Особенности списков, ссылки и клонирование.
    - Поиск элемента.
    - Матрицы.

Модуль 4

Функции

1. **Функции и модули.**
    - Что такое функция?
    - Цели и задачи функции.
    - Встроенные функции.
    - Математические функции и случайные числа.
    - Синтаксис объявления функций.
    - Аргументы и возвращаемые значения.
2. **Расширенные приемы по работе с функциями.**
    - Распаковка и упаковка аргументов.
    - Аргументы по умолчанию, аргументы-ключи.
    - Область видимости, правило LEGB.
    - Локальные и глобальные переменные в функциях.
    - Функции как объекты первого класса.
    - Рекурсия.
3. **Функциональное программирование.**
    - Что такое функциональное программирование?
    - Анонимные функции lambda.
    - Модуль functools.
    - Функции map(), reduce(), filter(), zip().

- Функции высших порядков.
- Замыкание.

4. **Замыкание.**
5. **Карринг.**
6. **Декораторы.**

Модуль 5

Сортировка, поиск

1. **Сортировка.**
   - Оптимальность.
   - Сортировка пузырьком.
   - Сортировка слиянием.
   - Сортировка Шелла.
   - Пирамидальная сортировка.
   - Быстрая сортировка.
2. **Поиск.**
   - Линейный поиск.
   - Бинарный поиск.

Модуль 6

Кортежи, множества, словари

1. **Кортежи.**
   - Коллекции неизменяемых объектов.
   - Применение и особенности кортежа.
2. **Множества.**
   - Математическое понятие множеств.
   - Тип данных set(), frozenset().
   - Операции над множествами.
   - Применение множеств.
3. **Словари.**
   - Ассоциативные массивы.
   - Хеш-таблицы.

- Создание словаря.
- Методы словаря.
- Понятие разреженной матрицы.

4. **Практические примеры использования.**

Модуль 7

Файлы

1. **Файлы.**
   - Файловая система, особенности реализации форматов.
   - Работа с файлами:
     - открытие;
     - закрытие;
     - чтение;
     - запись.
2. **Менеджер контекста.**
3. **Типы файлов, текстовые и бинарные.**
4. **Практические примеры использования.**

Модуль 8

Системы контроля версий

1. **Что такое контроль версий?**
2. **Зачем нужен контроль версий.**
3. **Обзор систем контроля версий.**
   - CVS.
   - SVN.
   - Git.
   - Другие системы контроля версий.
4. **Git.**
   - Что такое Git?
   - Цели и задачи Git.
   - Основные термины:
     - репозиторий;

        - коммит;
        - ветка;
        - рабочий каталог.
    - Операции с Git:
        - установка;
        - создание репозитория;
        - добавление файла в репозиторий;
        - запись коммита в репозиторий;
        - получение текущего состояния рабочего каталога;
        - отображение веток;
        - операции с накопительным буфером;
        - git remote;
        - git push;
        - git pull;
        - другие операции.
5. **Использование внешних сервисов (github).**

Модуль 9

ООП

1. **Введение в ООП.**
    - Понятие ООП.
    - Инкапсуляция.
    - Наследование.
    - Полиморфизм.
    - Особенности реализации ООП в Python, «утиная типизация».
2. **Типы данных, определяемые пользователем.**
    - Экземпляр класса.
    - Классы и объекты.
    - Поля (свойства), методы класса.
3. **Наследование и инкапсуляция.**
    - Общедоступный, внутренний и приватный метод.
    - Magic-методы, конструкторы.
    - Статические методы и методы класса.
    - Множественное наследование и MRO (порядок разрешения методов).

4. **Полиморфизм.**
   - Перегрузка операторов.
   - Реализация магических методов.
5. **Создание и управление поведением экземпляров класса.**
   - Функторы.
   - Декораторы.
   - Управляемые атрибуты.
   - Свойства.
   - Дескрипторы.
6. **Метаклассы.**
   - Модель метаклассов.
   - Метод конструктор __new__().
   - Протокол инструкции class.

Модуль 10

Структуры данных

1. **Связанные списки.**
   - Что такое список?
   - Односвязный и двусвязный список.
   - Практические примеры использования.
2. **Стек.**
   - Что такое стек?
   - Принцип LIFO.
   - Практические примеры использования.
3. **Очередь.**
   - Что такое очередь?
   - Виды очередей.
   - Практические примеры использования.
4. **Деревья.**
   - Что такое дерево?
   - Виды деревьев.
   - Практические примеры использования.

Модуль 11

Упаковка данных

1. **Сериализация и десереализация.**
   - Что такое сериализация?
   - Что такое десериализация?
   - Цели и задачи сериализации и десереализации.
   - Практические примеры использования.
2. **Модуль pickle.**
3. **Модуль json.**
4. **Сторонние модули сериализации.**

Модуль 12

Паттерны проектирования

1. **Что такое паттерны проектирования.**
2. **Причины возникновения паттернов проектирования.**
3. **Понятие паттерна проектирования.**
4. **Принципы применения паттернов проектирования.**
5. **Принципы выбора паттернов проектирования.**
6. **Принципы разделения паттернов на категории.**
7. **Введение в UML.**
   - Диаграмма классов.
   - Диаграмма объектов.
   - Диаграмма взаимодействия.
8. **Использование UML при анализе паттернов проектирования.**
   - Диаграмма классов.
   - Диаграмма объектов.
   - Диаграмма взаимодействия.
9. **Порождающие паттерны.**
   - Что такое порождающий паттерн?
   - Цели и задачи порождающих паттернов.
   - Обзор порождающих паттернов.
   - Разбор порождающих паттернов:

- Abstract Factory:
  - цель паттерна;
  - причины возникновения паттерна;
  - структура паттерна;
  - результаты использования паттерна;
  - практический пример использования паттерна.
- Builder:
  - цель паттерна;
  - причины возникновения паттерна;
  - структура паттерна;
  - результаты использования паттерна;
  - практический пример использования паттерна.
- Factory Method:
  - цель паттерна;
  - причины возникновения паттерна;
  - структура паттерна;
  - результаты использования паттерна;
  - практический пример использования паттерна.
- Prototype:
  - цель паттерна;
  - причины возникновения паттерна;
  - структура паттерна;
  - результаты использования паттерна;
  - практический пример использования паттерна.
- Singleton:
  - цель паттерна;
  - причины возникновения паттерна;
  - структура паттерна;
  - результаты использования паттерна;
  - практический пример использования паттерна.

10. Структурные паттерны.

- Что такое структурный паттерн?
- Цели и задачи структурных паттернов.
- Обзор структурных паттернов.
- Разбор структурных паттернов:
  - Adapter;
  - Composite;

- Facade;
- Proxy;
- другие структурные паттерны.

11. **Паттерны поведения.**
    - Что такое паттерны поведения?
    - Цели и задачи паттернов поведения.
    - Обзор паттернов поведения.
    - Разбор паттернов поведения:
        - Command;
        - Iterator;
        - Observer;
        - Strategy;
        - другие структурные паттерны.

Модуль 13

Паттерн MVC

1. **Что такое паттерн MVC?**
2. **Цели и задачи паттерна Model-View-Controller.**
3. **Model.**
    - Что такое Model?
    - Цели и задачи Model.
4. **View.**
    - Что такое View?
    - Цели и задачи View.
5. **Controller.**
    - Что такое Controller?
    - Цели и задачи Controller.
6. **Примеры использования паттерна MVC.**

Модуль 14

Принципы проектирования классов SOLID

1. **Обзор проблем, встречающихся при проектировании и разработке классов.**

2. **Принципы проектирования классов SOLID.**
   - Принцип единственности ответственности (The Single Responsibility Principle).
   - Принцип открытости/закрытости (The Open Closed Principle).
   - Принцип подстановки Барбары Лисков (The Liskov Substitution Principle).
   - Принцип разделения интерфейса (The Interface Segregation Principle).
   - Принцип инверсии зависимостей (The Dependency Inversion Principle).
3. **Примеры использования принципов SOLID.**

Модуль 15

Модульное тестирование

1. **Что такое модульное тестирование?**
2. **Цели и задачи модульного тестирования.**
3. **Необходимость модульного тестирования.**
4. **Обзор инструментов для модульного тестирования.**
5. **Инструмент для модульного тестирования Python-приложений.**

Модуль 16

Параллельное, многопоточное и сетевое программирование

1. **Параллельное и многопоточное программирование.**
   - Создание потоков.
   - Синхронизация потоков.
   - Очереди задач.
   - GIL и особенности реализации многопоточности в Python.
   - Процессы и передача данных между процессами.
2. **Сетевое программирование.**
   - Протокол HTTP/HTTPS.
   - Модель OSI, tcp/udp.
   - Клиент-серверная модель.
   - Реализация эхо-сервера простейшего, многопоточного и асинхронного.
   - Apache и Ngnix.

Модуль 17

Введение в работу с базами данных

1. **Введение в теорию баз данных.**
   - История и этапы развития.
   - Понятия база данных и система управления базами данных.
   - Сравнение существующих моделей баз данных:
     - файловая модель;
     - сетевая модель;
     - иерархическая модель;
     - реляционная модель;
     - объектно-ориентированная модель.
   - Понятие реляционной модели баз данных.
   - Двенадцать правил Кодда.
   - СУБД MySQL:
     - что такое MySQL;
     - история развития MySQL;
     - версии MySQL;
     - инсталляция MySQL.
   - Таблицы:
     - первичный ключ;
     - значение по умолчанию;
     - уникальность.
   - Типы данных.
   - Индексы.
   - Запросы:
     - введение в язык структурированных запросов SQL;
     - язык SQL. Стандарты языка SQL;
     - понятия DDL, DML, DCL.
2. **Запросы SELECT, INSERT, UPDATE, DELETE.**
   - Оператор SELECT:
     - предложение SELECT;
     - предложение FROM;
     - предложение WHERE;
     - предложение ORDER BY.
   - Ключевые слова IN, BETWEEN, LIKE.
   - Оператор INSERT.

- Оператор UPDATE.
- Оператор DELETE.

3. **Многотабличные базы данных.**
   - Аномалии взаимодействия с однотабличной базой данных:
     - аномалии обновления;
     - аномалии вставки;
     - аномалии обновления;
     - аномалии удаления.
   - Принципы создания многотабличной базы данных:
     - причины создания многотабличной базы данных;
     - внешний ключ;
     - связи. Типы связей;
     - целостность данных.
4. **Нормализация.**
   - Необходимость нормализации.
   - Понятие нормальной формы.
   - Первая нормальная форма.
   - Вторая нормальная форма.
   - Третья нормальная форма.
   - Нормальная форма Бойса-Кодда.
5. **Многотабличные запросы.**
   - Принципы создания многотабличного запроса.
   - Декартовое произведение.
6. **Функции агрегирования.**
   - Функция COUNT.
   - Функция AVG.
   - Функция SUM.
   - Функция MIN.
   - Функция MAX.
7. **Понятие группировки. Ключевое слово GROUP BY.**
8. **Ключевое слово HAVING. Сравнительный анализ HAVING и WHERE.**
9. **Подзапросы.**
   - Необходимость создания и использования подзапросов.
   - Сравнение подзапросов и многотабличных запросов.
   - Принцип работы подзапросов.

10. **Операторы для использования в подзапросах, объединения.**
    - Операторы для использования в подзапросах:
        - оператор EXISTS;
        - операторы ANY/SOME;
        - оператор ALL.
    - Объединение результатов запроса:
        - принципы объединения;
        - ключевое слово UNION;
        - ключевое слово UNION ALL.
    - Объединения JOIN:
        - понятие inner join;
        - понятие left join;
        - понятие right join;
        - понятие full join.
11. **План выполнения запроса.**
12. **Оптимизация запросов.**
13. **Понятие транзакции. Использование транзакций.**
14. **Представления.**
    - Создание представлений.
    - Модификация представлений.
    - Удаление представлений.
    - Изменения данных через представления.
15. **Хранимые процедуры.**
16. **Триггеры.**

Модуль 18

Использование баз данных в Python

1. **Использование баз данных.**
2. **ORM системы.**
    - Теория и практика использования.
    - SQLAlchemy.
    - PonyORM и другие.

3. **NoSQL базы данных.**
   - Основы NoSQL, теорема CAP.
   - Основные типы NoSQL баз данных.
   - СУБД Redis.
   - СУБД MongoDB.
4. **Работа с базами данных, сериализация данных.**
   - Запись и чтение в формате XML DOM.
   - StAX и SAX-парсеры.

Модуль 19

Работа в команде, управление программными проектами

1. **Что такое управление программными проектами?**
2. **Причины возникновения дисциплины управление программными проектами.**
3. **Диаграммы Ганта.**
4. **Важные вопросы по управлению программными проектами.**
   - Что такое проект и программный проект?
   - Что такое жизненный цикл процесса разработки программного обеспечения?
   - Что такое управление проектами?
   - Что такое одиночная разработка?
   - Что такое командная разработка?
   - Анализ проблем одиночной и командной разработки программного обеспечения.
5. **Анализ терминов предметной области.**
   - Процесс.
   - Проект.
   - Персонал.
   - Продукт.
   - Качество.
6. **Характеристики проекта.**
   - Тип проекта.
   - Цель проекта.
   - Требования к качеству.

- Требования к бюджету.
- Требования по срокам завершения.

7. **Расходы, связанные с проектом.**
   - Прямые.
   - Непрямые.
8. **Общий обзор моделей и методологий процесса разработки.**
   - Фазы процесса:
     - определение требований;
     - проектирование;
     - конструирование («реализация», «кодирование»);
     - интеграция;
     - тестирование и отладка («верификация»);
     - инсталляция;
     - поддержка.
   - Водопадная модель.
   - Спиральная модель.
   - Итеративная модель:
     - Agile;
     - Scrum;
     - XP.
   - RUP.
   - MSF.
   - Анализ существующих моделей и методов.
9. **Подробнее о Scrum.**
   - Что такое Scrum?
   - Причины возникновения Scrum.
   - Роли в Scrum:
     - владелец продукта;
     - команда;
     - scrum-мастер.
   - Бэклог продукта:
     - что такое бэклог продукта;
     - как создавать бэклог;
     - как оценивать задачи в бэклоге;
     - что такое scrum-доска;
     - примеры создания бэклога.

- Спринт:
  - что такое спринт;
  - планирование спринтов;
  - ежедневный скрам;
  - обзор спринта;
  - ретроспективное собрание.

Практическое задание

Необходимо провести симуляцию работы команды по методологии Scrum. Например, это может быть так называемое скрам-лего.
Подробно тут: Scrum Simulation with LEGO Bricks.

Модуль 20

Фреймворки

1. **Классификация web-фреймворков.**
   - Web-фреймворк Flask.
   - Механизм шаблонов и язык шаблонов Jinja2.
   - Сессии и формы.
   - WEB-фреймворк Bottle.
2. **Асинхронные web-приложения.**
   - Фреймворк Tornado.
   - Библиотека Twisted.
3. **Django – фреймворк для создания web-приложений.**
   - Установка Django.
   - Создание проекта Django.
   - Структура Django-проекта (url-view-model-template).
   - Паттерн MVC/MVT.
4. **Модели и ORM.**
   - Модели и поля.
   - Связи между таблицами, проектирование и реализация БД благодаря механизму ORM.
   - Миграции.
   - Менеджеры модели.
   - Административная часть.
5. **Работа с административной частью, настройка отображения.**
   - Настройка интерфейса администратора.

- Сортировка, фильтры, редактирование полей.
- Вывод данных.

6. **Язык шаблонов и создание web-форм.**
   - Шаблоны.
   - Роутинг, представления.
   - Синтаксис, логические конструкции.
   - Контекстный процессор.
7. **Создание форм и страниц.**
   - Добавление страниц.
   - GET/POST запросы.
   - Статическое содержимое и динамическое.
   - Авторизация.
   - Ограничения прав доступа.
   - Валидация.
   - AJAX.
8. **Стандартные задачи.**
   - Встроенные class-based views.
   - Использование форм с Django CBV.
9. **Стандартные задачи.**
   - Аутентификация.
   - Пагинация.
10. **Погружение в Django.**
    - Middleware.
    - Сигналы.
    - Сообщения.
    - Сессии.
11. **Погружение в Django. Продолжение.**
    - Древовидные структуры в Django.
    - Миксины.
    - Дебаггер Django.
    - Логирование, отправка почты.

Модуль 21

Создание чат-ботов с помощью Python

1. **Что такое чат-бот?**
2. **Цели и задачи чат-ботов.**
3. **Архитектура чат-бота.**
4. **Практический пример создания чат-бота.**

Модуль 22

Экзамен